LEARNING TO UNDERSTAND NEW FACIAL EXPRESSIONS

A Thesis

by

PALASH PARMAR

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Anxiao Jiang
Committee Members,	Theodora Chaspari
	Chao Tian
Head of Department,	Dilma Da Silva

August 2019

Major Subject: Computer Science

ABSTRACT

Facial expression recognition is getting popular in the research community because of its extensive use in understanding human sentiments. Among various medium of human interaction uses in daily life, the facial expression is the most direct form of communication that explains a lot about human emotions. Because of this reason, researchers are actively exploiting this field of human-computer interaction.

The research aims for the development of automatic facial expression annotation for context-based database generation. We pointed out the limitation of an existing facial expression detection system for real-world application and studied new ways to bridge current research and user application. We proposed a one-shot learning-based automatic facial expression labeling technique which requires very few manual labels to understand the context of sentiment in expression and utilizes them to train facial expression system with a specific use case. The evaluation of the proposed model is done with two methods (i) we manually labeled few more examples and tested the model against those examples, and (ii) from the seven basic facial expressions, we kept one facial expression separate and used those example to test the efficiency of the model.

DEDICATION

To my parents

ACKNOWLEDGMENTS

I would like to express the deepest appreciation to my thesis advisor, Professor Anxiao (Andrew) Jiang, who has the attitude and the substance of a genius, without his guidance and persistent help, this thesis would not have been possible.

I would also like to thank my committee members, Professor Theadora Chaspari and Professor Chao Tian for serving the committee.

In addition, a thank you to my friends Vineet Garg and Arnav Kundu for sharing their knowledge and involvement in intense discussion with me on deep learning and computer vision which improved my knowledge and understanding of the subject. I also thank Shubhangi Gupta, who always motivated me and believed in me. I am grateful to all my family and friends in College Station and back in India for their constant support and encouragement.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a thesis committee consisting of Professor Anxiao (Andrew) Jaing and Professor Theadora Chaspari of the Department of Computer Science and Engineering and Professor Chao Tian of the Department of Electrical and Computer Engineering.

Funding Sources

There are no outside funding contributions to acknowledge related to the research and compilation of this document.

NOMENCLATURE

FER	Facial Expression Recognition
FACS	Facial Action Coding System
AU	Action Unit
YOLO	You Look Only Once (object detection model)
FAP	Facial Animation Parameters
CNN	Convolutional Neural Networks
R-CNN	Region Convolutional Neural Networks
NN	Neural Networks
SSD	Single Shot Detector
LBP	Local Binary Patterns
HOG	Histogram of Oriented Gradients
IOU	Intersection over Union
FPS	Frames per Second
GUI	Graphical User Interface

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION AND LITERATURE REVIEW

There has been a lot of research going on developing artificial machine intelligence for understanding human. The scope of such research is extensive covering speech recognition, text summarization, machine translation, action recognition, etc. Out of which, facial expression recognition is an essential form of communication that human uses and hence the need for developing such a system are self-explanatory [4, 5]. Since 1971, when Ekman and Friesen [6] proposed methods to quantify primary emotion into six unique facial expressions, several new advancements have been witnessed.

Initially, the facial expression is primarily the domain of psychologists, but an investigation by Suwa et al. [7] in 1978 on automatic facial expression from image sequence attracted many computer science researchers in this domain. Another factor for facial expression gaining attention is advancements accomplished in related research areas such as face detection, tracking, and recognition. One thing to note here is most of the studies are done on facial expression in an attempt to understand human emotions, but there is no direct linkage between facial expression and human emotions [8, 3]. This creates the foundation of our research by introducing one crucial factor, Context to facial expression.

Till now, many state-of-the-art models have been proposed which can correctly recognize human facial expression in a complex and challenging environment. But all such model can recognize basic facial expression due to the limitation of context-based facial databases. This is the reason such models are good on paper but lacks real-life application. For example, neutral expression means differently in different scenarios. Human perceives emotion from expression by combining contextual information. This motivates me to think beyond the scope of basic facial expressions and come up with a method to generate broad context dependent facial databases.

While the end goal of the research is coming up with more efficient Facial Expression Recognition (FER) system which suits to real life application, the initial proposal of the study is a new technique to harness context-aware dataset for development of such system. The research is in-

spired by the work of Brenden et al.'s One-shot learning [9] in learning deep models using a few examples. Some focus is also given on another component of the FER system such as face detection, expression recognition, and FER system deployment.

For developing such a FER system, I propose to use deep learning methods for detection, recognition and database development. Deep neural network techniques recently yielded impressive results across a variety of tasks and competitions including ImageNet [10] and hence encouraged me to research further in this methodology.

1.1 Facial Parameterization

Facial expression recognition research started taking formal shape when researchers in Psychology parameterized facial behaviors and motions. Before such advancement, researchers rely on human observation which is often prone to error and subject to one's perception. Till now, we have three successful attempts to represent the facial expressions which a machine can understand. Almost all the research in recognition of facial expression uses one of the three (based on its advantages and disadvantages). They are as follows:

1.1.1 Categorical Model

Presented by Ekman et al. [6], face is assigned an expression from a universal set of emotion constant across all cultures. Such parameterization takes the whole face into account while deciding facial expression. Further, Ekman proposed six universal forms of expression (namely Happy, Sad, Anger, Fear, Surprise, Disgust) which are easy to distinguish and form the base for any complex facial expression.

Figure 1.1: Categorical model examples, CK+ Dataset [1]

1.1.2 Dimensional Model

In such a model, an expression is described by a set of independent dimensions or emotional scales such as Arousal, Distress and Valence [2]. These dimensions are on a continuous scale. Here Valence is referred to as "how positive or negative the effect of emotion is" and arousal measures "the magnitude of emotion." Russell proposal the use of emotional scales as Valence and Arousal but any orthogonal parameter which can represent the human emotion can be used in this model. However, arousal and valence are widely used.

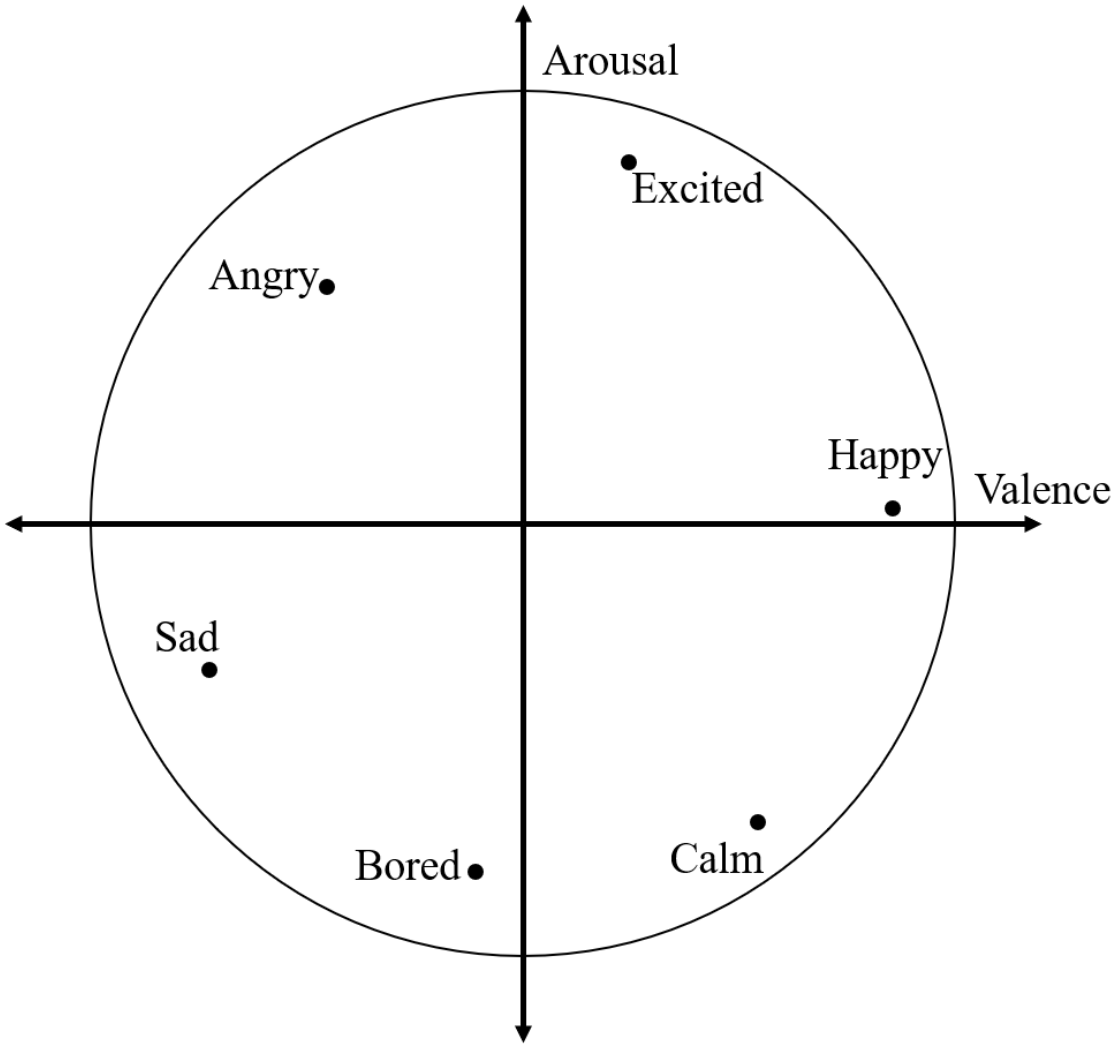
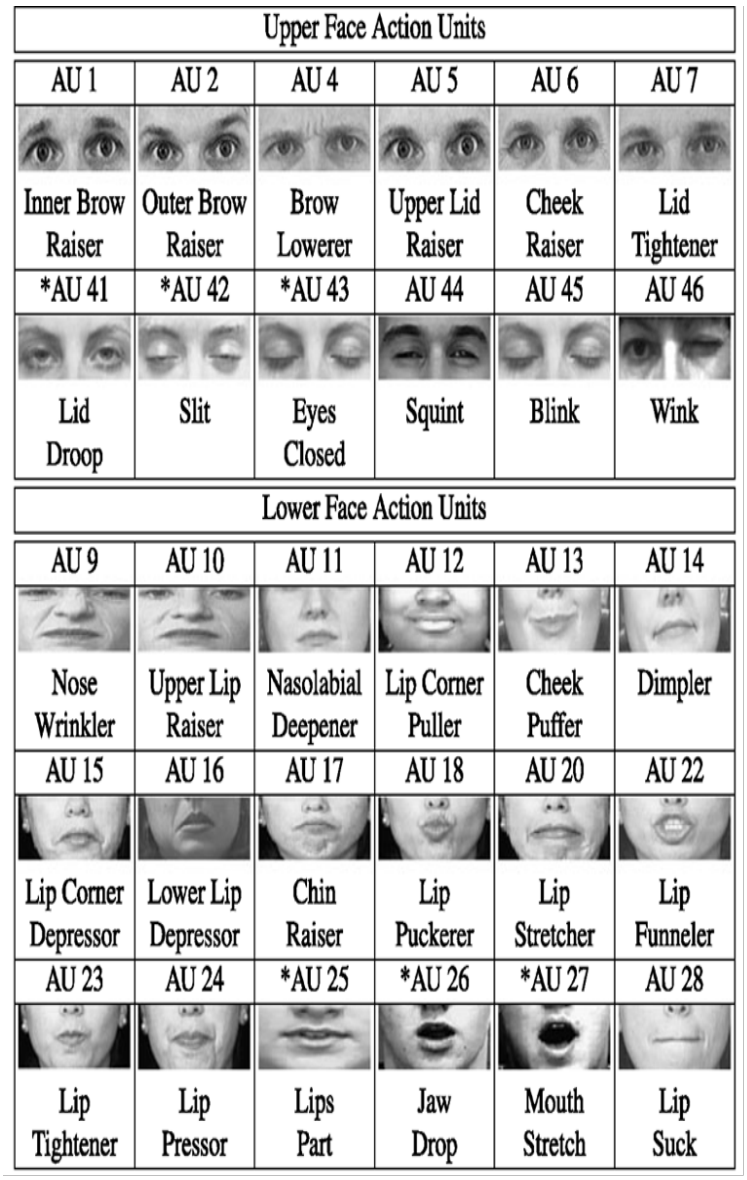


Figure 1.2: Circumplex model of effect [2]

1.1.3 Facial Action Coding System (FACS)

Facial Action Coding System [3] are codes assigned to a specific gesture of facial muscles such as eyes, nose, and mouth. Humans, while showing a particular expression, display a set of code for each of facial muscles. Different combination of muscle movement leads to a wide range of facial expression. These muscle codes are called Action Units (AU), and these AUs do not represent facial expression directly. Some of the Action units are shown in the figure.



(a)

Basic expressions	Involved Action Units
Surprise	AU 1, 2, 5,15,16, 20, 26
Fear	AU 1, 2, 4, 5,15,20, 26
Disgust	AU 2, 4, 9, 15, 17
Anger	AU 2, 4, 7, 9,10, 20, 26
Happiness	AU 1, 6,12,14
Sadness	AU 1, 4,15, 23

(b)

Figure 1.3: Facial Action Coding System [3], (a) representation of Action Units, and (b) Relation between AUs and expression

1.1.4 Facial Animation Parameters (FAPs)

This is a relatively new and better version of FACS. The motivation of this model comes from MPEG-4 facial animation [11] where graphics researchers are more focused on facial movement rather than muscle caused that expression. This concept is further formalized by Cowie et al. [12] to represent facial expressions as location and motion of landmark points on face as shown in the figure. According to Cowie, while MEPG-4 are related to animation and synthesis of facial expression, they are strongly related to AUs which defines the basis of FACS (page 125 of [12]).

1.2 Facial Databases

Facial Databases are a crucial aspect of the development of the FER system. It not only validates the FER system but also used in training of FER deep learning model. A system can only achieve an accuracy as high as the facial database. Table 1.1 shows some widely used databases for training and testing FER models. There are several factors for classifying different dataset. They are

- Facial parameterization model used
- Spontaneous or posed
- controlled environment or random
- frontal face posed or random facial position

Dataset	Description	Subjects	Condition	Expression Mode
CK+ [13]	- Frontal face	- 123	- Lab - Posed	- 7 categorical emotion - 30 AUs
MultiPie [14]	- More than 750,000 images - 15 viewpoint and 19 illuminations	- 337	- Controlled -Posed	- 7 categorical emotion
MMI [15]	- 2900 videos and high resolution images	- 75	- Controlled - Posed & Spontaneous	- 31 AUs - Six basic expression
AFEW [16]	- Videos extracted from movies	- 330	- Wild	- 7 categorical emotion
FER-2013 [17]	- Web crawled images	- 35,786	- Wild	- 7 categorical emotion
EmotioNet [18]	- Web crawled images - 100,000 images annotated manually - 900,000 images annotated automatically	- 100,000	- Wild	- 12 AUs annotated - 23 categorical emotion based on AUs
Aff-Wild [19]	- Videos extracted from YouTube	- 500	- Wild	-Valence & Arousal
FER-Wild [20]	- Web crawled images	- 24,000	- Wild	- 7 categorical emotion
Affect-Net [21]	- Web crawled images	- 450,000	- Wild	- 8 categorical emotion - Valence & Arousal

Table 1.1: Summary of existing facial expression databases

There are several datasets publicly available in which subject are recorded in controlled environment [1, 15, 22, 14] and many FER model has been proposed which provide excellent results on such databases. However, work done by Sebe et al. [23] pointed out the limitation of such databases. They are

- Different subject show the same expression with different intensities
- Authenticity of expression loses as soon as the subject being aware that he has been photographed
- Authenticity of expression also loses even if the subject is not pictured, but the feeling is not spontaneous

If defining a robust FER model, it should have excellent performance with any facial pose, spontaneous expression in an environment with real-life problems such as occlusion and severe lighting conditions. The robustness of deep neural model relies heavily on how the dataset is

generated. AffectNet [21] facial expression database is the best fit into all the circumstances. It is a web collection large corpus of facial images which are manually labeled into 9 categories (Neutral, Happy, Sad, Surprise, Fear, Disgust, Anger, Contempt, None, Uncertain, Non-Face). The face parameter model provided with this dataset is the categorical model, dimensional model, and FAPs model.

Earlier databases are either lab generated [13] [14] where subjects are required to present certain expression or manually labeled [17] [18] [19] making database generation a complicated and time consuming task. It is the same reason we only see databases with basic expression. Mollahosseini *et al.* [21] and Benitez-Quiroz *et al.* [18] worked on automation of database generation but their techniques uses existing database to generate a generalized model for labeling unlabeled faces into basic expression only.

1.3 Image registration for FER

Object detection works at the base of many computer vision system. By identifying the subject in the image, further recognition activities become much more straightforward. For example, in face recognition, the task is much easier if we can get the bounding box of a face. Many neural network object detector present now does the job of regression and detection simultaneously. Face detection is a particular but challenging case of general object detection algorithm because of face pose, occlusion and lighting conditions.

Face detection algorithm can be broadly categorized as appearance-based, template matching based, feature-based and knowledge-based methods [24]. Researchers proposed various techniques under these four categories, but a breakthrough in face detection is seen after 2000 when Viola and Jones [25] proposed a practical face detector which can run in real time. Though this detector can detect a frontal face, their work motivated several other works.

The present face detection algorithm is highly inspired by the deep learning approach because of its outstanding performance in another vision task. While the basis of many modern face de-

tection system originates from a generalized object detection algorithm, it is beneficial to mention these work in that order. Most popular 2-stage object detector available are R-CNN [24], Fast R-CNN [26] and Faster R-CNN [27] network. Though 2-stage detector has significantly higher detection accuracy than another detector such as YOLO [28], running such a model is still computationally expensive with Faster R-CNN to achieve five fps. Next class of detectors are single shot detectors which includes YOLO [28], YOLO9000 [29], YOLOv3 [30] and Single Shot MultiBox Detector (SSD) [31]. Such class of detector provide reasonable accuracy but demand comparatively less computation power. YOLOv3 can achieve up to 45 fps.

Some of the recent developments in deep learning have specifically focused on face detection in images. Faceness-Net [32] is an approach which generates separate feature map for different parts of the face such as eyes, hairs, nose, mouse, etc. followed by creating face proposals and reranking bounding box by face measure. Another work by Cheng Chi et al. [33] introduces a Selective Refinement Network, (SRN), which reduces the false positive rate and improve location accuracy simultaneously by a Two-step Classification and Two-step Regression module. Extending Cheng's work, Shifeng Zhang et al. [34] improved SRN network by introducing new augmentation technique, improved backbone network.

Most of the modern face detection techniques are a neural network based and require labeled face database. There are mainly two databases that are widely used for training face detection models, (i) FDDB [35] and (ii) WIDER FACE [36] database. Out of the two, WIDER FACE is more comprehensive, and big database and all the results are generated against WIDER FACE testing dataset.

1.4 Facial Expression Recognition

Facial Expression recognition is a popular research topic nowadays, and more and more researchers are contributing their work in this field. Table 1.2 summarized some of the facial expres-

sion recognition techniques. All the mentioned techniques have cutting edge performance on the datasets mentioned in table 1.1

Initial FER algorithm used conventional image processing algorithm for detecting expression in face. Earlier databases mainly consist of lab controlled posed frontal face (as mentioned in section 1.2), and the conventional algorithm is sufficient to address this non-complex recognition task. These algorithm uses hand-picked features and hence are hard to generalize on faces in the wild. Some of the recognized algorithm uses Gabor wavelet filtered whole-face [37], LBP [38], and HOG [39].

Advancement of deep neural network methodologies is also seen in FER systems mainly because of the availability of large databases and computing capabilities which makes these algorithms highly generalizable in wild settings. Some of work using deep learning techniques are [40], [41], and [42]. Details of all algorithms are presented in table 1.2.

FER technique	**Dataset**	**Description**	**Accuracy**
Mollahosseini *et al.* [40]	CK+ MultiPie	- GoogleNet based CNN	- CK+ ⊣ 93.2% - MultiPie ⊣ 94.7
Shan *et al.* [38]	MMI	- LBP features with SVM RBF classifier	- MMI ⊣ 86.9%
Fan *et al.* [41]	AFEW	-CNN-RNN and 3D hybrid network - Trained on both Images and Videos	- AFEW ⊣ 56.16%
Tang *et al.* [42]	FER-2013	- CNN & SVM cascaded architecture	- FER-2013 ⊣ 71.2%
Benitez-Quiroz *et al.* [18]	EmotioNet	- Gabor wavelet features trained with - KSDA classifier	- EmotioNet ⊣ 80%
Mollahosseini *et al.* [20]	FER-Wild	- AlexNet model for classification - noise estimation methods used	- FER-Wild ⊣ 82.12%

Table 1.2: Best performing facial expression recognition technique on different dataset

2. FACE DETECTION

2.1 Methodology

Neural network based techniques are widely popular because of its generalization and performance. We define a face detection task as identifying face bounding box in an image. Mathematically, the algorithm will provide all set of face position (x, y) and dimensions (w, h) for all the faces present in an image. If we define such a model as ϕ, we can write

$$(f_1, f_2, ..., f_n) = \phi(image)$$

where f_i is a set of (x, y, w, h) of face in an image where x is the position of face on width axis, y is the position of face on height axis, w is the width of face and h is the height of the face. As face detection lies under object detection category, there are specific requirements that are particular to our use case that we followed while designing a face detection model. They are

1. We have targeted faces with width resolution of at least 10% of overall image resolution because we might lose too much information for Facial Expression Recognition. Keeping face resolution of 10% ensures that the model can work with images with variable sizes leading to generalization and better fit.

2. We gave equal priority to detection accuracy and detection speed because face detection in cascade with FER will significantly affect the overall inference time of the system.

3. Face detection algorithm is targeted at detecting at-most 5 faces in an image. Keeping limited faces ensures better resolution for each face. Since, we are giving equal priority to speed and accuracy, limiting the number of faces balances the trade-off.

We are proposing to use anchor based convolutional network-based model ϕ, because of its real-time capabilities and decent detection accuracy. Anchors are initial detection boxes the algorithm uses to localize the object. Such an algorithm produces a large number of bounding boxes

for each spatial location in the image. Each bounding box is associated with the probability of an object's presence. We can filter the actual object by threshold the object's probability. Such methodology makes the model differential end-to-end which detecting multiple objects. More details about algorithm and anchor are shown in the following sections. We modified the YOLOv3 [30] object detection model based on the requirements mentioned above.

2.2 Architecture

Architecture like Faceness-Net [32] and SRN [33] are very efficient in detection face (including the tiny face in the image) but computationally very expensive and has prolonged detection rate. Because of this reason we used YOLO architecture.

2.2.1 Backbone

The overall network architecture is shown in figure 2.1. The network takes a color image of dimension 416×416. Each layer in the network is represented by operation name, kernel size, and filter dimension. Each box also represents the current feature size of an image by the side of the box. There are residual links in the network inspired from darknet [30] architecture. Each residual layer ends with the addition of an existing layer with the incoming residual unit making it an identity mapping [43]. Each YOLO block output the information of the detected object. In YOLOv3 [30], YOLO block is applied at 3 different scales feature in model, (i) at 13×13 (responsible for object detection of large size), (ii) at 26×26 (responsible for object detection of medium size), and (iii) 52×52 (responsible for object detection of small size). Since we are limiting face size to large and medium, we only use YOLO block at feature scale 13×13 and 26×26 (as shown in figure 2.1). In conclusion, the base model reduces the spatial dimension of the input image to 13×13 and 26×26 and forward it to the respective YOLO block.

Each YOLO block is responsible for the detection object in the spatial feature map provided by the backbone model. For each grid in the spatial space, the block creates multiple anchors (initial bounding box of a face). Detailed explanation on anchors is provided in section 2.4. Further, each anchor provides information about the position of the object with respect to grid position

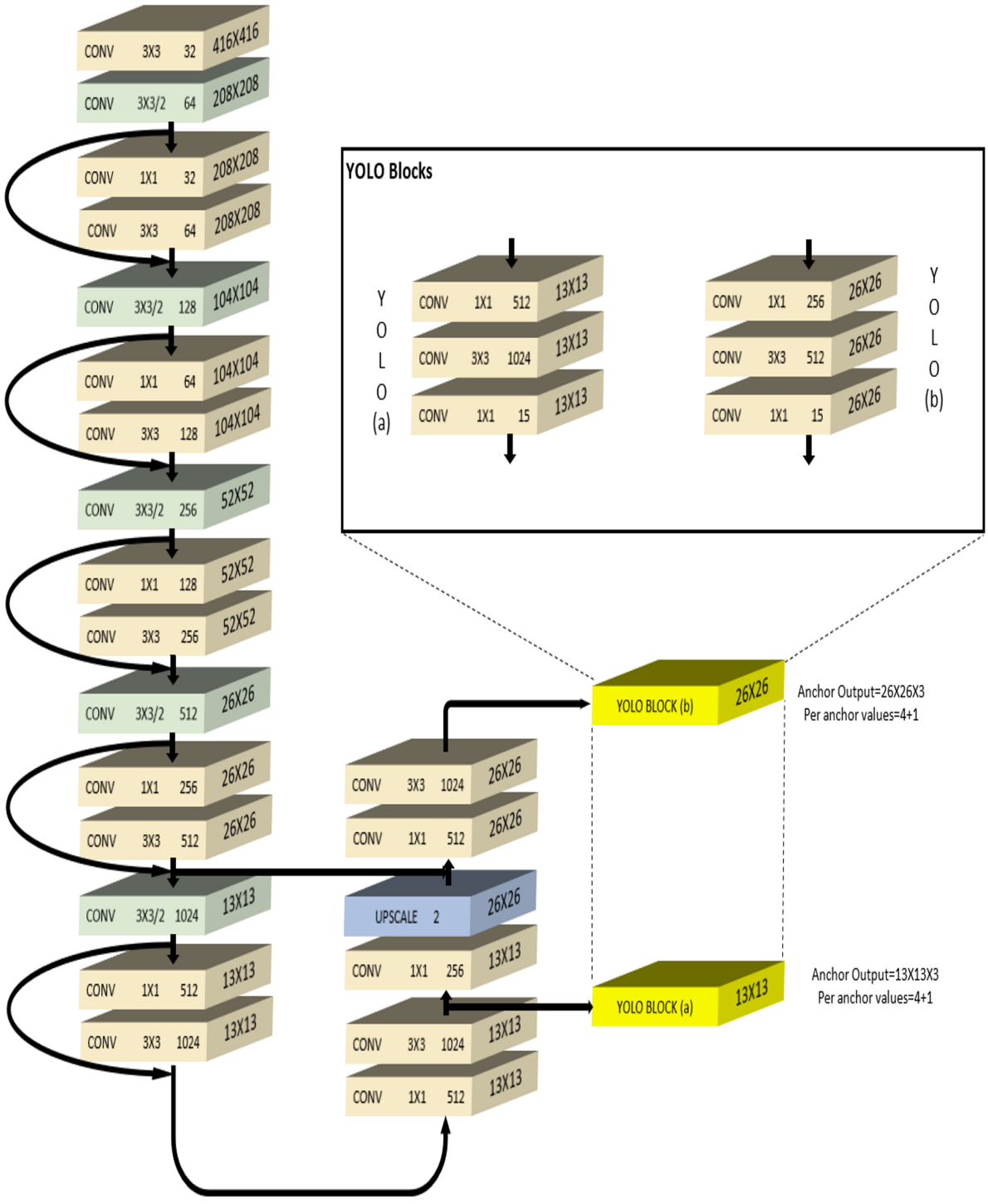


Figure 2.1: Face detection architecture: Network takes 416×416 color image for detecting face. Each box represent a layer in the network with layer operation detail in front and layer 2D shape in the side. Network output 5 values for each 1×1 feature in the YOLO layer, 4 for x, y, w, h corrections on initial anchor box and last value is the probability of face in that anchor.

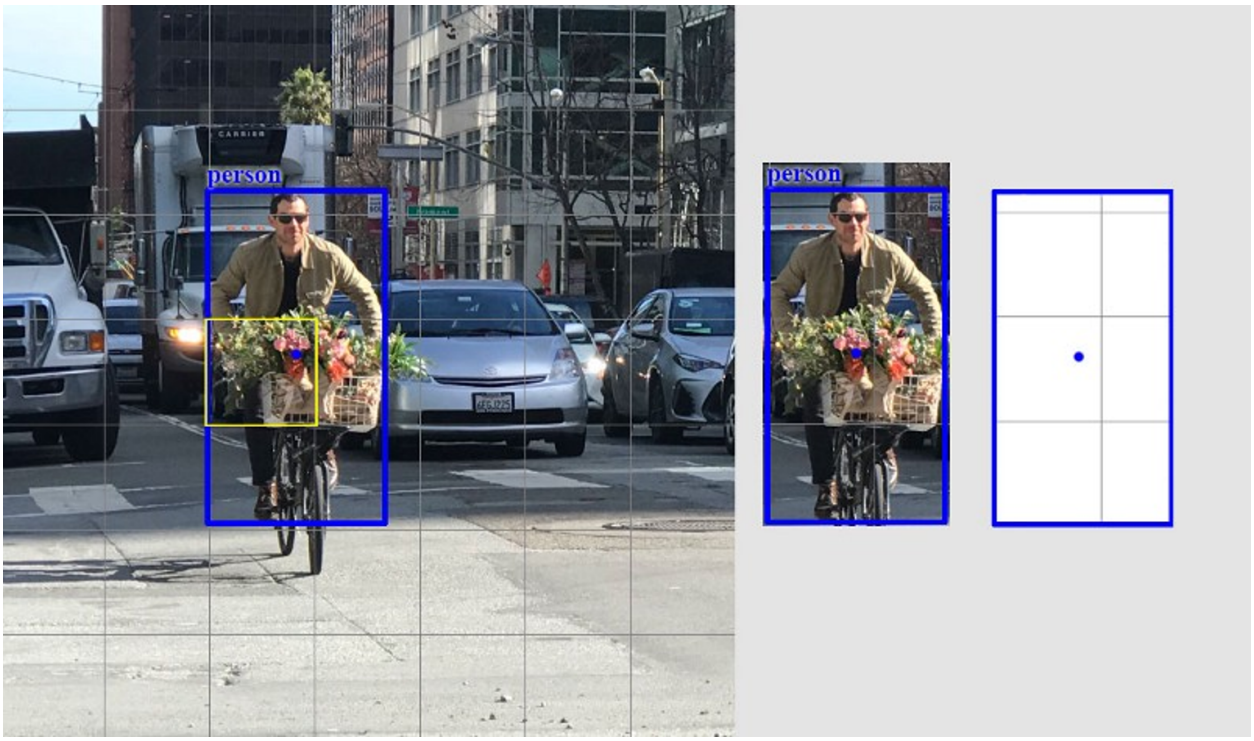
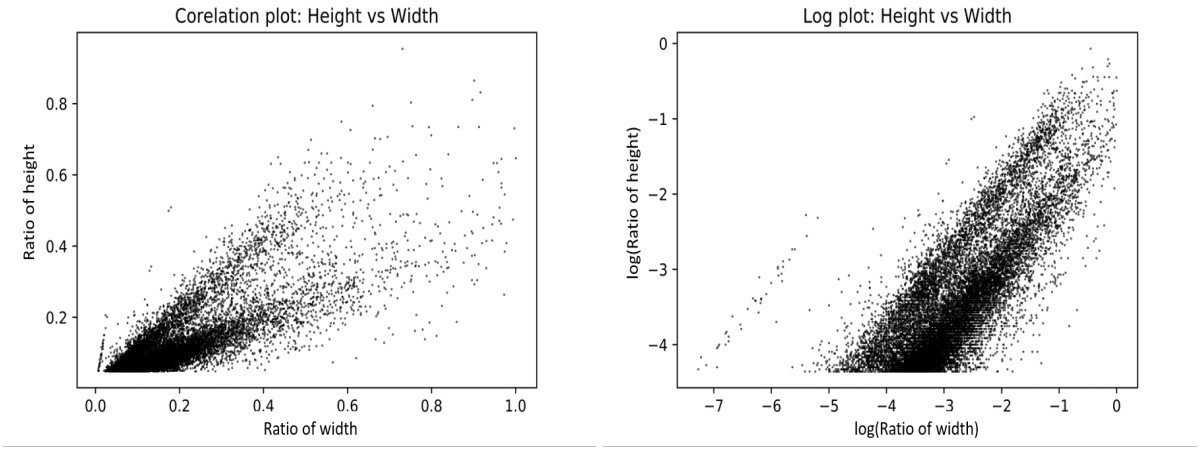


Figure 2.2: Grid representation in spatial space. YOLO block generate anchors for each grid representing an object which output x, y, w, h and object confidence

and confidence with which the object is present in the grid as shown in figure 2.2. In another way, with each grid's feature size, we will get information about the relative position of object and probability of an object's presence. This way the YOLO block provides a fixed number of an object irrespective of how many objects present in the image. Anchor with the actual object can be filtered out using a probability score of each object.

2.2.2 YOLO block

This section discusses the details of YOLO block shown in figure 2.1. Each grid in the spatial feature map at different scale comprises of B bounding boxes for each grid, and each bounding box has output x, y, w, h and confidence score for object presence. With such schematic, if we take 3 bounding boxes per grid ($B = 3$), the output of YOLO block at scale 13×13 will be 13×13×15 (15 = 3 bounding boxes × 5 (x, y, w, h, and confidence)). Similarly, the output of YOLO block at scale 26×26 will be 26×26×15. For each image, the algorithm will output a total of 2535



(a) (b)

Figure 2.3: Height vs. width plots of faces from WIDER FACE dataset for understanding correlation between them. (a) plot of ratio of face height to image height vs. ratio of face width to image width, (b) plot of log of face height vs image height vs. log of ratio of face height to image width

bounding boxes ((13×13×) + (26×26×3)).

2.2.3 Anchor Design

An anchor point is a very crucial part of any convolutional based object detector. The bounding boxes discussed in section 2.2.2 provides detected face in terms of its position and size. The position provided by the model is relative to some initial bounding boxes (fixed initial box the each grid takes for fine fit the object) that the model takes. Such relative prediction is easier to learn and provide good accuracy. The initial bounding boxes are called anchors. It is suggested that closes the shape and position of initial bounding box (anchor), less amount of regression needs to learn by the neural network and results in higher accuracy. Hence, choice of anchors are critical in object detection algorithm. As discussed in previous section, each grid in the spatial space can have multiple bounding boxes, similarly for each bounding box, we have different anchor from which the YOLO block predict the output bounding box and confidence.

For specific object detection task (i.e., face), we can calculate anchor which proves better initial bounding boxes. Human face poses a specific aspect ratio as opposed to a random object. This is

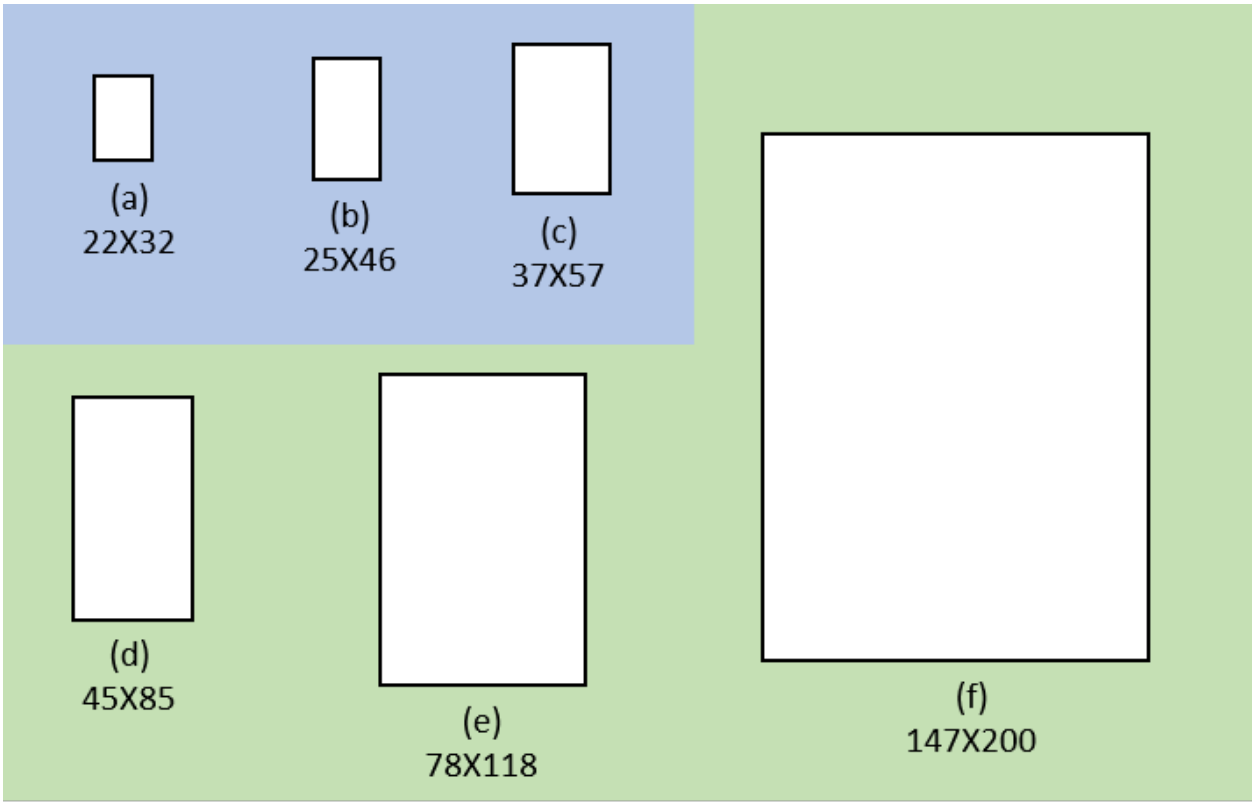


Figure 2.4: Probable anchor sizes with respect to image 416×416. (a), (b) & (c) are smaller anchors and used for YOLO blocks at scale 13×13 and (d), (e) & (f) are larger anchors for YOLO block at scale 26×26. These prior anchors covers IOU of 75%. Initial IOU of 75% with these anchors are good start point for the face detector and final IOU after detection will improve upon it.

further justified by plotting ratio of face width to image width and face height to image height from the training dataset as shown in figure 2.3. It can be concluded that there exists some anchor size which is more dominant in the dataset that other and also there exist a strong relationship between width and height suggesting the presence of specific anchor ratio. Inspired by Joseph Redmon *et al.* YOLO9000 [29], we run k-means on training dataset for finding best priors anchors. We use IOU metric for clustering instead of Euclidean metric. IOU (Intersection Over Union) is the ratio of the area of intersection of predicted object and ground truth to the area of the union of predicted object and ground truth. Higher the IOU suggest that predicted bounding box is very closer to the ground truth bounding box. This metric is independent of the size of bounding box making it an

unbiased evaluation metric than Euclidean distance based metric. The number of cluster vs. IOU is shown in figure 2.5. Increasing the number of groups will increase the detection accuracy but will result in increasing the number of convolutional filters in the output of YOLO block (refer section 2.2.2). For example, if we increase the number of anchors (bounding boxes for detection) per grid from five to ten, the resultant filter output of YOLO block at scale 13×13 will increase from 13×13×25 ((4+1)×5) to 13×13×50 ((4+1)×10). This increase in filter size will add extra burden on neural network in detecting a face.

From the plot of cluster vs. IOU in figure 2.5, taking six anchor boxes (clusters) cover an IOU of 75% on the training dataset. It suggests that without applying any correction on initial anchors, the object detector can achieve 75% IOU. Figure 2.4 shows the candidate anchor boxes with size anchor configuration.

2.2.4 Loss Function

The face detection algorithm is performing two tasks, (i) predicting correct x, y, w and h of faces relative to anchor (regression) shown in equation (2.1), and (ii) calculating the probability of object present in the predicted box (classification) shown in equation (2.2). The loss functions for both classification and regression are used the same as YOLOv3. Because we are dealing with a single class (face), the classification loss reduces to just the bi-linear cross entropy loss (probability of the presence of face). The overall loss function after modification is shown below.

$$\begin{aligned} Loss_{regression} =& \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ &+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \end{aligned} \tag{2.1}$$

$$Loss_{classification} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{noobj} (C_i - \hat{C}_i)^2 \tag{2.2}$$

In equation (2.1), we use sum-squared error for regression loss for position and size of face.

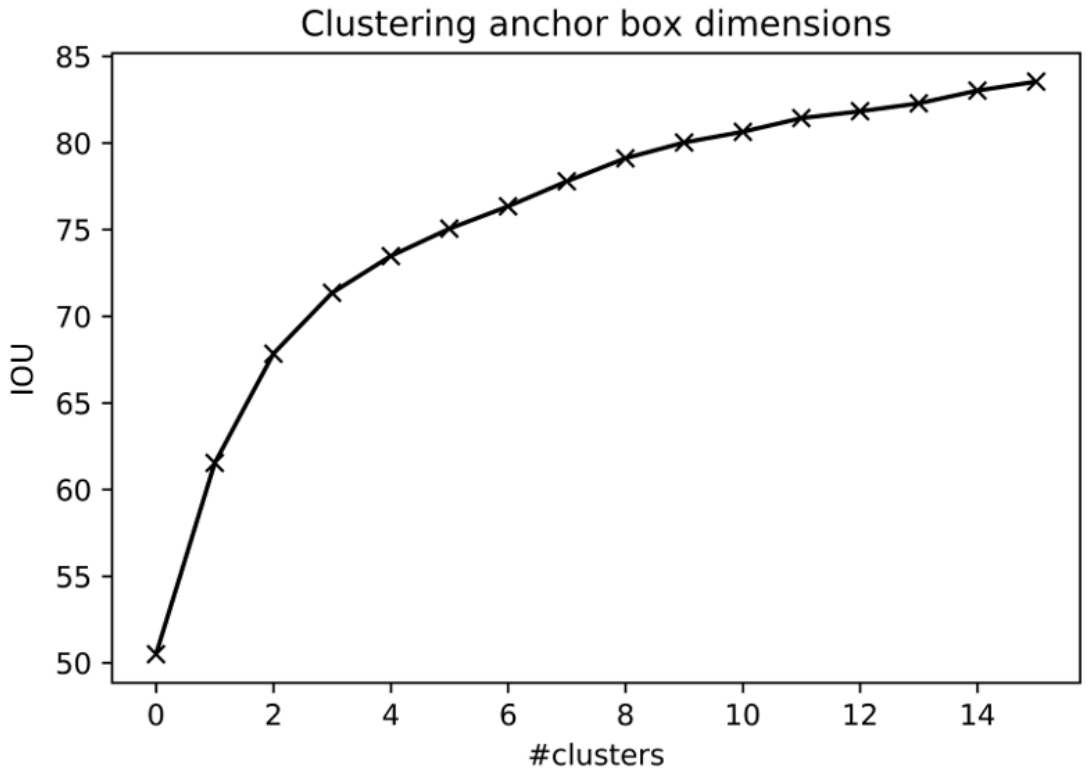


Figure 2.5: IOU achieved with number of clusters. Here clusters are different probable anchors (face height and width) as features. For six anchor (clusters), the IOU of anchors with dataset faces is 75%

x_i, y_i, w_i and h_i are ground truth bounding boxes and $\hat{x}_i, \hat{y}_i, \hat{w}_i$ and $\hat{h}_i$ are predicted position and size of bounding boxes. Loss function uses square-root of width and height as it should penalize less for small deviations on large bounding boxes than small boxes. Also, $1_{ij}^{obj} = 1$ if the j_{th} bounding box in cell i is responsible for detecting the face, otherwise 0. The parameter λ_{coord} control the weight of regression loss in overall loss function. In equation (2.2) shows the confidence loss of an face present in box and confidence score of face not present in a box (mentioned as obj and $noobj$). $1_{ij}^{obj} = 1$ if j_{th} bounding box in cell i is responsible for detecting the face, otherwise 0. 1_{ij}^{noobj} is complementary of 1_{ij}^{obj} representing face is not present in the box (background). $\hat{C}_i$ is the confidence score of box j in cell i.

2.2.5 What did not work

Various experiments are done by modifying the darknet network. In section 2.2.1, we mentioned the placement of YOLO block at the end of network architecture at two different spatial scales (i.e., 13×13 and 26×26). We also experimented with using three YOLO blocks at three scales (at 13×13, 26×26, and 52×52). YOLO block at scale 52×52 results in addition of extra 40560 (52×52×15) bounding boxes. This YOLO block is responsible for detecting face with very small dimension. Since we restricted the dimension of the face, such addition of YOLO block does not improve the face detection accuracy. But the addition of extra 40560 bounding boxes add significant computational burden on network and inference time becomes worst. Next, we experimented with changing all the Convolutional Neural Network to Depthwise Separable Convolutional Neural Network [44]. Depthwise Separable CNN has roughly 2.3 times less computation over CNN. Depthwise Separable CNN demonstrated much-reduced computation requirement keeping the classification accuracy same on ImageNet [45] classification challenge when compared to other state-of-the-art models. We also replaced all the CNN in the model architecture to Depthwise Separable CNN with the intend to improve inference time keeping detection accuracy same. We observed a marginal decrease in inference time but the detection accuracy severely affected (reduced). We observed that Depthwise Separable CNN do not perform in all cases because it reduces the number of parameters in convolution and if your network is already small, it results in poor performance. The result of the experiments is summarized in the following sections.

2.3 Training and Inference

2.3.1 Training Dataset

Training of face detection model is done on WIDER FACE [46] dataset. It comprises of 32,203 images of different dimensions and 393,703 annotated face bounding boxes with a different pose, scale, occlusion, and lighting condition. The dataset is divided into training (80%), validation (20%).

2.3.2 Data Preprocessing

WIDER FACE contains faces scaling from very small to large. We are only targeting the face dimension of at least 10% hence we removed face annotation with size less than 10%. Dataset also contains images with different sizes, but the proposed model can only accept images with size 416×416. If the image is perfect square (width is equal to height), then we resize the image to 416×416 using bi-linear interpolation. If the image is not a perfect square, we make it perfect square by padding the shorter side to make it equal to longer side using a constant 127.5 (color range from 0-255) and then apply resize function. We also recalculated the annotation based on the new image shape and size.

2.3.3 Optimization

The loss function for the model is taken as the summation of equation (2.1) and equation (2.2). The network is trained end to end, with weights initialized with He initialization [47]. We trained the model with ADAM with momentum 0.9, weight decay 0.0005 and batch size 32. The learning rate is set to 10^{-3}. The two YOLO blocks return the probable bounding boxes, their confidence and the loss value required for optimization during training as discussed in section 2.2.2. The model is trained for 100 epochs. Training statistics are shown in figure 2.6. Figure 2.6 (a) plots the overall loss with respect to training epochs and figure 2.6 (b) provides the plot for accuracy, precision and recall with respect to epochs. The whole model is implemented in PyTorch library [48].

2.3.4 Inference

During inference time, the image is preprocessed as shown in section 2.3.2. In evaluation mode, the model returns just the final bounding boxes masking the other outputs. We filter these bounding box with a confidence score (probability of face in bounding box) greater that a threshold (0.5). Such operation will only keep bounding boxes with face in it and discard other boxes. At this point, for a single face object, the confidence score (probability of face in bounding box) is higher for all the surrounding boxes, but all these boxes are providing localization for the same object. The duplicate boxes are further filtered by applying Non-Maximal Suppression (NMS).

NMS takes the bounding box with highest confidence score among all the other boxes, check if any other box is intersecting the selected box more than a threshold, it removes that box as it is the bounding box for the same face. Applying NMS to the bounding box generate detection boxes for unique faces.

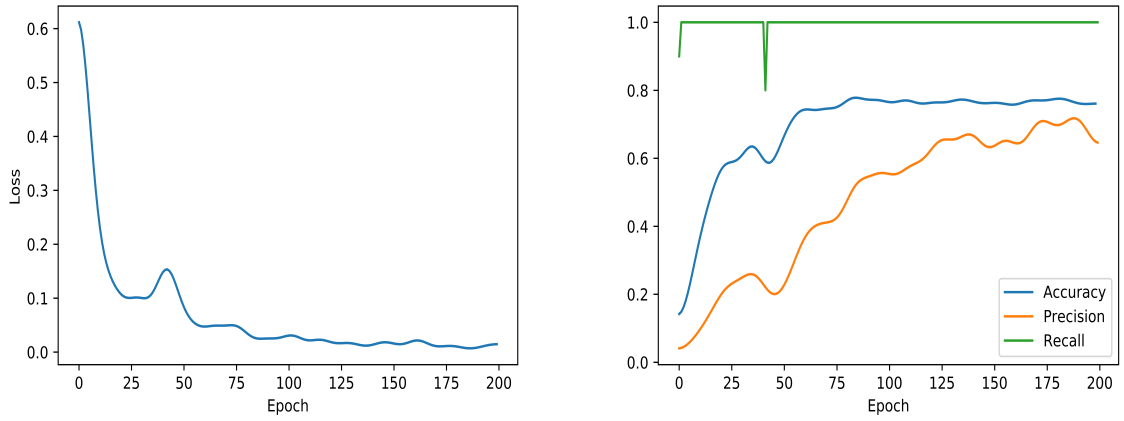
2.4 Experiments and Results

We experimented with different modified versions of the model but accounted for four important ones. We compared the models by Precision-Recall curves, average precision and inference time. All the models are tested against the WIDER FACE validation set. Benchmark testing is done against YOLOv3 model.

The Precision vs. Recall curve on the WIDER FACE validation set is shown in figure 2.7. Higher the area under Precision vs. Recall curve, better the face detector is. The results of average precision, inference time and FPS are shown in table 2.1. YOLOv3 performance is best among all other models but the inference time is inferior. This is justified from the fact that it is a bigger model with vast learnable parameters. Our proposed face detection model with two YOLO blocks is runner up in terms of performance but has a significant advantage over inference time. It outperforms the same architecture with three YOLO block. This might be because it will be resulting in false positive faces at small scale because of extra YOLO block.

At last, our assumption of Depthwise Separable Convolution in face detection does not work well. Though it has improved inference time because of less learnable parameters, its performance is unsatisfactory. Hence, Depthwise Separable Convolution's performance is highly situational and does not produce good results in all situation.

Figure **??** shows the output of the face detector on some of the randomly collected images from validation dataset. The proposed face detector can detect face with unposed faces, occluded faces, faces in poor lighting condition and blurred faces in the background.



(a) (b)

Figure 2.6: Experiment outcome: (a) loss vs. epochs, & (b) testing accuracy, precision and recall vs. epochs

	Average Precision	Inference Time	FPS
YOLOv3	0.873	23.22 ms	43.06
Depthwise Separable Convolution	0.635	15.28 ms	65.44
Proposed model with 3 YOLO blocks	0.778	18.57 ms	53.85
Proposed Face Detector with 2 YOLO blocks	**0.823**	**16.03 ms**	**62.37**

Table 2.1: Face detection results summary

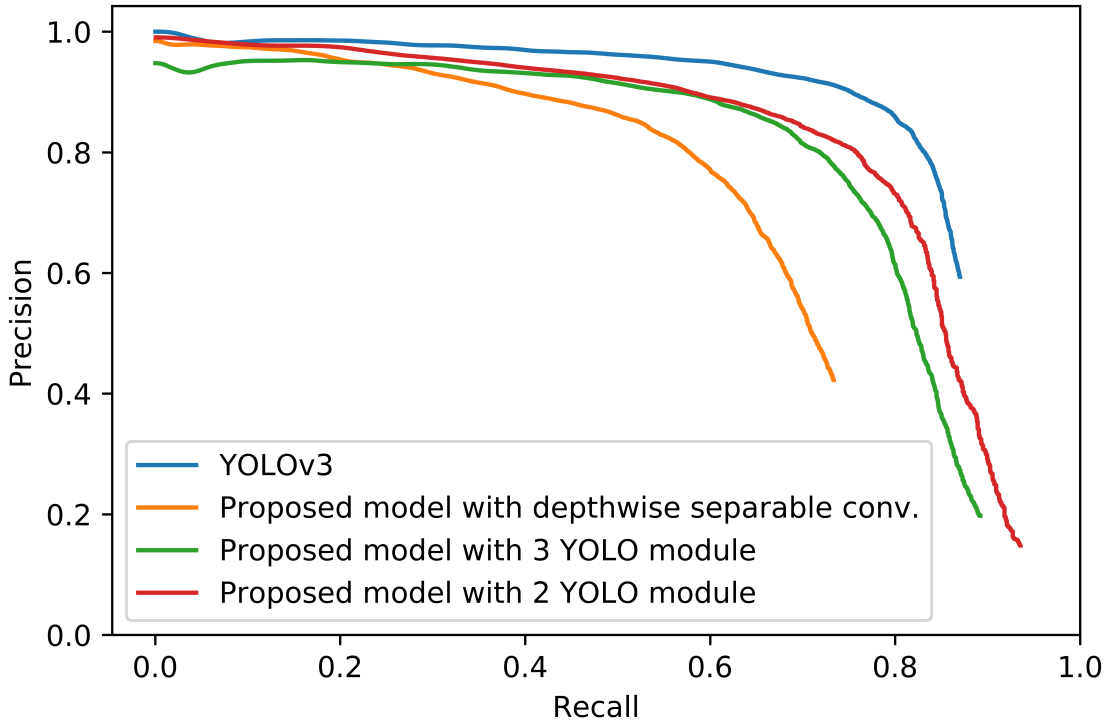
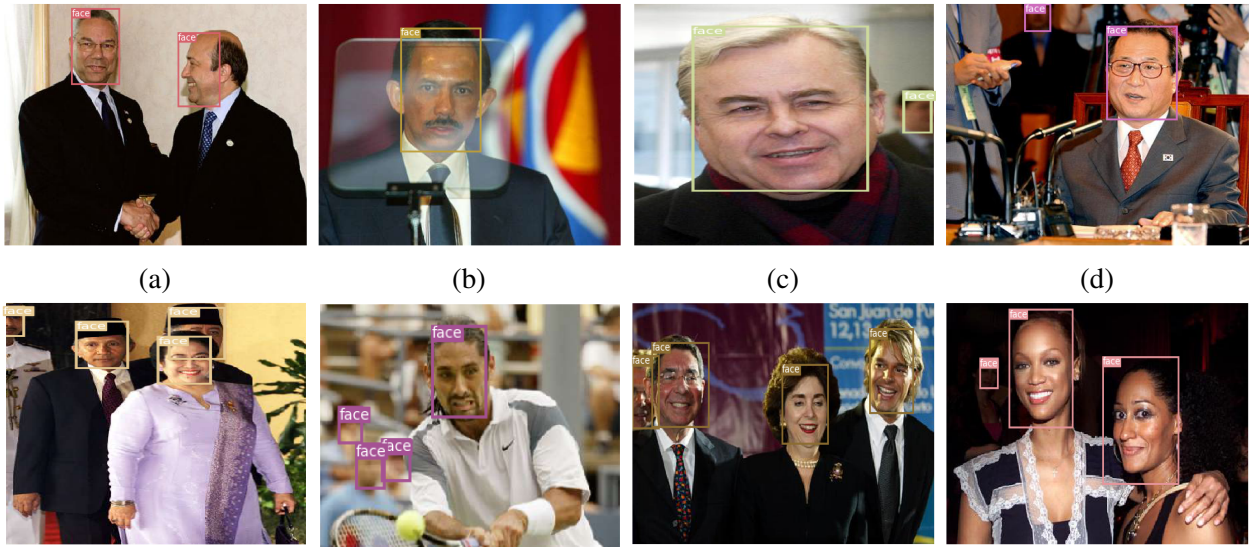


Figure 2.7: Experiment Results: Precision vs. Recall curves



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2.8: Face detection results from proposed method. (a) demonstrate complicated pose, (b) and (g) demonstrate occluded face detection , (d), (e) and (h) demonstrate small and blurred face detection

3.1 Problem Statement

With this work, we define the task of labeling new facial expression with only a few examples. We employed few-shot learning based meta-learning technique where we have three datasets, (i) training set which is the copious amount of labeled dataset on basic (known) facial expression, (ii) is the context set (support set) which are few examples training set for new facial expression, and (iii) testing set which will have same classes as that of context set and use to test the final model. A few shot learning problem can also be interpreted as C-way K-shot if the context set contains K labeled examples for each of C unique classes.

Since we are interested in new facial expression labeling, we can train a classifier using context set which can assign a facial expression class $\hat{y}$ for the samples x in the testing set. However, such classifier would not perform well because there are very few examples in a context set to train it with. Therefore we trained a meta-network on a training set, to extract transferable knowledge

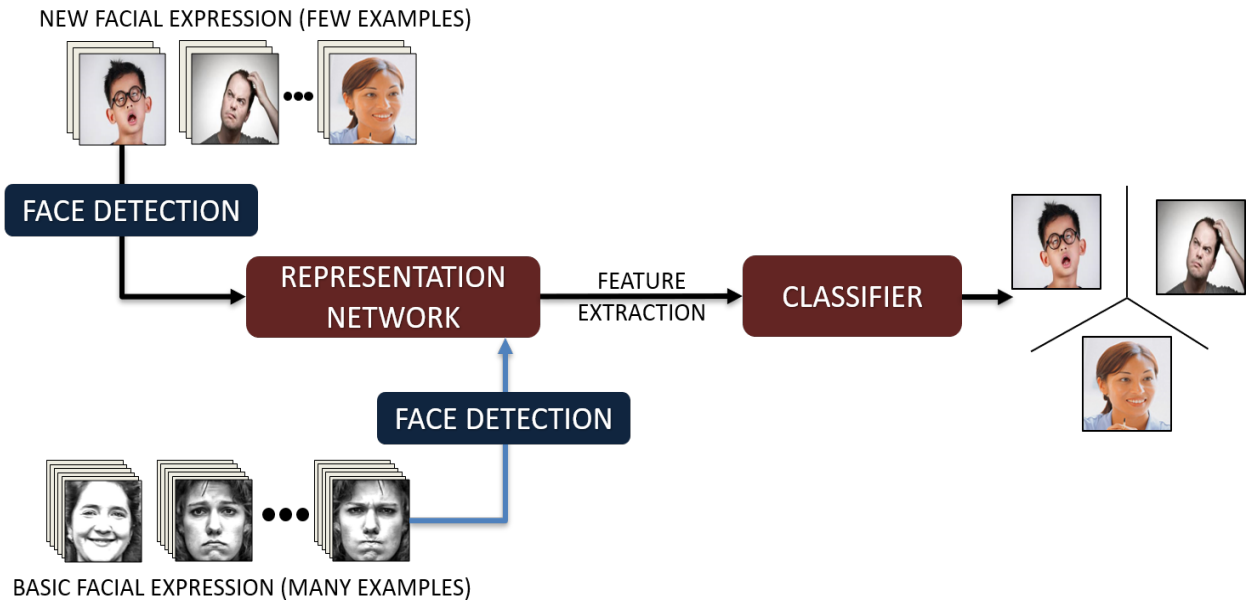


Figure 3.1: Our few shot learning outline

which allows us to train a strong classifier on a few context set.

3.2 Representation Network

We solve the problem of understanding new facial expression by few-shot learning. We divide the methodology into two parts which are described in the following subsections. First, we train a neural network known as Representation Network ϕ which is capable of generating a representation of subtlety in small features of facial expression in the form of a feature vector. Second, we represent any face (facial expression) in the form of discriminative feature vector using Representation Network and use these features to train a strong classifier W to classify similar expression in new faces. Mathematically, we use context set S to define a classifier W which define a mapping $S \rightarrow W$. Here classifier is a function which takes feature vector from representation network and predicts the new facial expression class.

3.2.1 Architecture

We define a deep neural network known as Representation Network ϕ which can be trained on the existing copious amount of training data which has a different class of expression than the context set. After training, Representation Network will act as an excellent feature extractor and allow the effective classifier to be trained with a few examples. Intuitively, our goal is to train these network in such fashion that it generates a similar feature for similar expression and different features for different expression.

We choose VGG net [49] as our base model for Representation Network. We modified VGG19 model as shown in figure 3.2. For robust learning and better generalization, we added batch normalization layer [50] and RELU layer after all convolutional layers. All the convolutional layers have a kernel size of 3 and padding 1. Intermittent max pool layers in between convolutional layers have kernel size 2 and stride 2. Last convolutional layer in the VGG network is flattened into a fully connected layer with ReLU activation function. Another fully connected layer of size 64 is applied after the flatten layer which acts as an embedding layer and provides a feature vector for the face image. Last fully connected layer is the logit layer that is used for training representation

network with few facial expressions. Network output both class logits and feature vector for both training and testing.

While there is no direct established relation for determining feature vector size, there are specific evidence and observation that proved useful in determining the feature vector size to be 64. It is observed that the number of neuron in a layer reduces as the layer get closer to output layer making the network a funnel-shaped structure. This is because a neural network keeps the abstracted information useful for predicting the correct answer while discarding unwanted information from the input data. We follow the same paradigm for designing fully connected layer in representation network. Flatten layer consist of 128 neuron and last output layer consist of 7 neurons. For efficient training, it is evident the embedding layer should be of 64 (less than 128 and greater than 7). Some more work by Jeffrey *et al.* [51] for developing word embedding suggest that embedding size depending on how broad spectrum the vector is covering. One more observation of the experiment with different embedding size done by Jeffrey is that the information represented by the embedding vector increases with vector size only up to a certain extent and saturate afterwords. Such observation motivated us to use 64 as a safe feature vector size. However, analyzing different feature vector size is out of the scope of work for this time because the new facial expression spectrum is vast.

3.2.2 Representation Network Loss Function

For training representation network, basic facial expressions such as happy, sad, angry, surprise are used. With these labels available in the dataset, we apply cross entropy classification loss function to train the representation network. However, such loss function only makes the Representation Network's feature vector good on these few labels but does not make the network's features a good discriminator against new facial expressions. This effect can be seen by visualizing the feature space of a feature vector, but it is not possible to visualize a 64-dimensional vector. For visualizing the effect of the classification loss function, we trained the same representation network by making the embedding layer size equal to 2. By restricting the feature size equal to two, we can observe the features of the 2D space. Figure 3.3 (a) shows the plot for modified 2D features

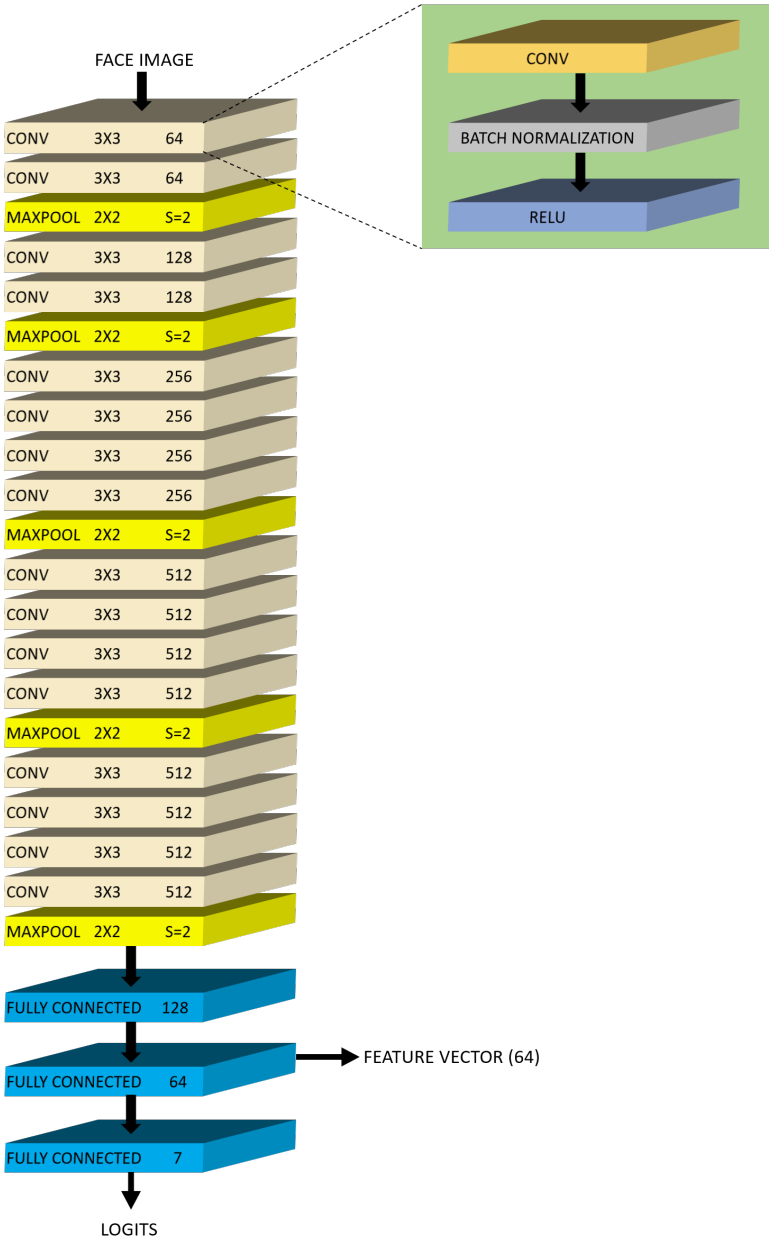
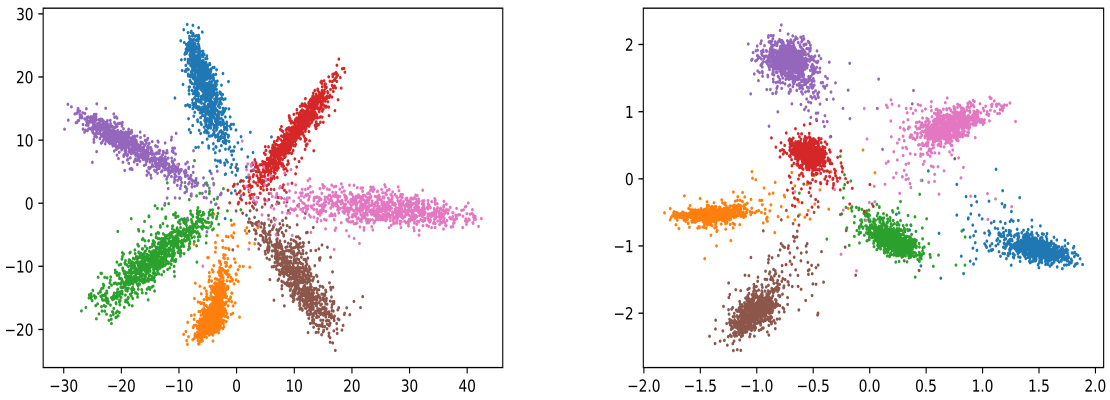


Figure 3.2: Representation Network architecture: Network takes the input image size of 300×300. Each block represent the operation used in the network with kernel size and filter size mention along with it. Fully connected layer with 128 neuron is non-linear layer with ReLU and rest are linearly activated layer. Output layer is activated using soft-max function for converting network output to probabilities.



(a) (b)

Figure 3.3: Feature visualization: (a) cluttered feature with cross-Entropy loss function, (b) improved features due to cross entropy + contrastive center loss

collected for testing data. We can observe that the features are clustered at the center of the plot for all the classes.

To improve these, we added a contrastive-center loss [52] proposed by Qi *et al.* along with the classical cross entropy loss function. The overall loss function is mentioned below:

$$H_{y'}(x, y) := -\sum_{i} y'_i \log(y_i) + \frac{1}{2}\sum_{i=1}^{m} \frac{||x_i - c_{y_i}||_2^2}{(\sum_{j=1, j\neq y_i}^{k} ||x_i - c_j||_2^2) + \delta} \tag{3.1}$$

where y is the true expression and y' is the predicted expression. x is the feature vector from the representation network, c is the class center for feature vector and chosen randomly, m is the number of examples in a mini-batch, and δ is a number to make denominator non-zero. The class center c is updated through the training process by back-propagating using center loss function as shown in equation (3.1). Such an update will penalize the cluster centers being too close to make them discretely distributed in features space. The loss function also works in such fashion that it penalizes similar class features being apart, and also different class features being similar. After application of this loss function, the features become more discriminative as seen in figure 3.3 (b).

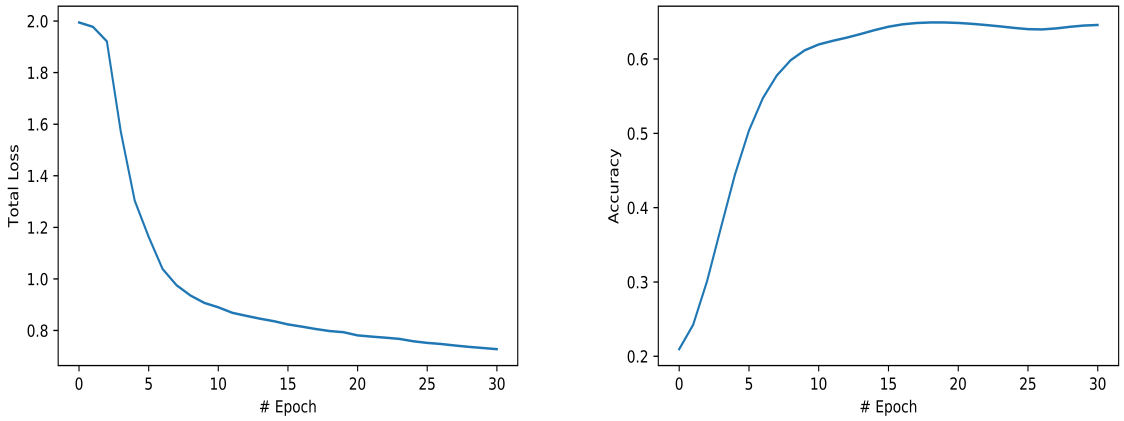
3.2.3 Training and inference

For training the representation network, we used Affect-Net database, which contains 280,000 manually labeled images and 450,000 automatically annotated images in 7 different facial expressions. They are *Neutral, Happy, Sad, Angry, Surprise, Contempt and Disgust*. The dataset is split into training and validation set by randomly sampling 500 images from each expression category making a 3500 example testing set. The database is highly imbalanced, and we employed weighted loss function while training the model. Weight for each label is proportional to the inverse of the number of example of that label in the database with a constraint that sum of all weights should be 1. Face localization coordinates are provided in the database and used to extract the face form image. For training, we randomly chose a corner in the localized face and cropped a part of an image from that corner making a face little off-center along the chosen corner. Such technique improves the generalization of neural network training. Next, the cropped image is resized to 300×300 and normalized it before feeding it for training. Normalizing is done by subtracting the image from its mean value and diving by the standard deviation to make it zero mean and unit deviation. Labels are marked from 0-6 representing 7 facial expressions. At inference or testing time, the face image is just resized to 300×300 and normalized.

The model is trained on 4 GPUs with synchronized training. The effective batch size becomes 100 and trained for 30 epochs. The learning rate is chosen to be 0.001 for initial 5 epochs. After 5 epochs, the learning rate is reduced by 0.9 for every 5 epochs. We also employed gradient clipping to 0.1 for stable learning. The network achieves a final accuracy of 64% on test set. Figure 3.4 shows the loss curve and accuracy curve with respect to epochs.

3.3 Classifier

Representation network is trained to generate discriminative features for the newer facial expression. The last stage of getting annotation for new facial expression is the classification of the feature vector for the corresponding expression. There are many available algorithms that can be used for classifying expression from features. The criteria for choosing any algorithm is how



(a) (b)

Figure 3.4: Training Statistics: (a) Loss curve wrt. epochs, (b) accuracy curve wrt. epochs

few examples are needed to train the classifier for recognizing new facial expression and what maximum performance can be achieved.

In general, machine learning algorithm such as K-Nearest Neighbors, Support Vector Machine, Random Forest and Logistic Regression works well with few-data points than a deep neural network. Also, these algorithms are fast to train and faster in predicting also. So, we experimented with these machine learning algorithm for classification of new facial expression.

3.4 Overall Pipeline

This section presents the work-flow of the overall technique connecting all the modules shown above (pipeline shown in figure 3.1). The pipeline can be divided into three phases. Firstly, the representation network is trained with existing facial expression databases with limited facial expressions (such as Affect-Net [21]) as shown in section 3.2.3. Such a trained network works as a good discriminator in generating a feature vector for any emotion. This phase is required to be done just for one time.

In the second phase, the end classifier algorithm is trained with few labeled support set using the features generated by the representation network. Here, the model learns the context required for new facial expression in the form of example faces provided by the user for training. This

training phase has to be repeated each time a new context has to be provided.

Lastly, with trained representation network and classifier trained on provided support set, the network takes an unlabeled face and label it using classifier based on feature vector generated by the representation network. This phase is repeated for the rest of the unannotated images for labeling.

3.5 Evaluation

Evaluation of methodology is critical as it not only shows how better the overall model is in recognizing the new facial expression but also say that how well the representation network is in discriminating subtle facial expression features in the face. As there is not specific dataset available for testing new facial expression, we evaluate our model in two cases as mentioned below:

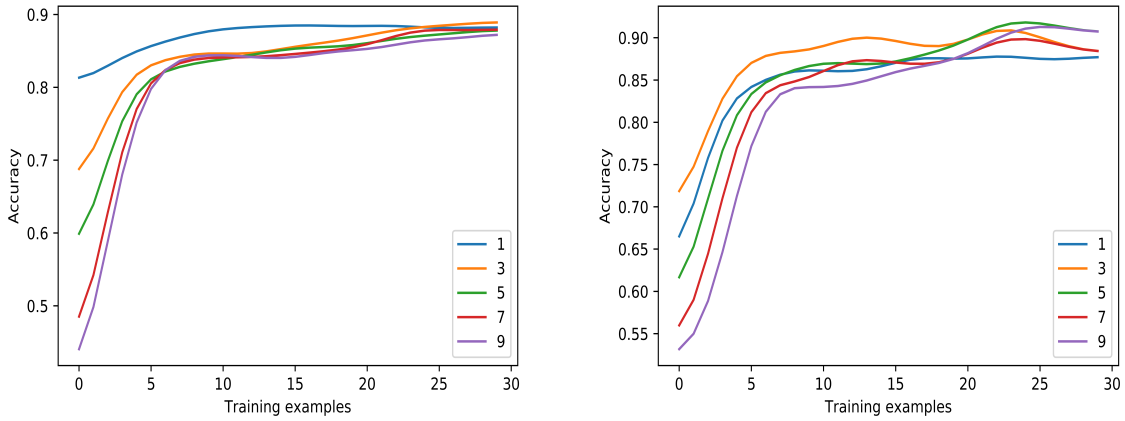
1. We used the existing unseen database for facial expression recognition with basic expression to test the model. CK+ facial expression dataset is used for this experiment.

2. We manually label 150 images for new facial expression and utilize this dataset for testing. This labeling is manually done by human by choosing two categories, i.e. positive and negative and labeled with a context of *people's experience at a grocery store*.

In both cases, we utilize a few randomly chosen examples (1-30) from the dataset as a training example for the classifier and rest and testing examples. Since performing evaluation by choosing a few training examples is highly biased as accuracy will depend on which examples are randomly chosen, we performed 1000 fold experiment. This means we randomly selected training examples (different all the time) and repeated this 1000 times to get a unbiased performance numbers.

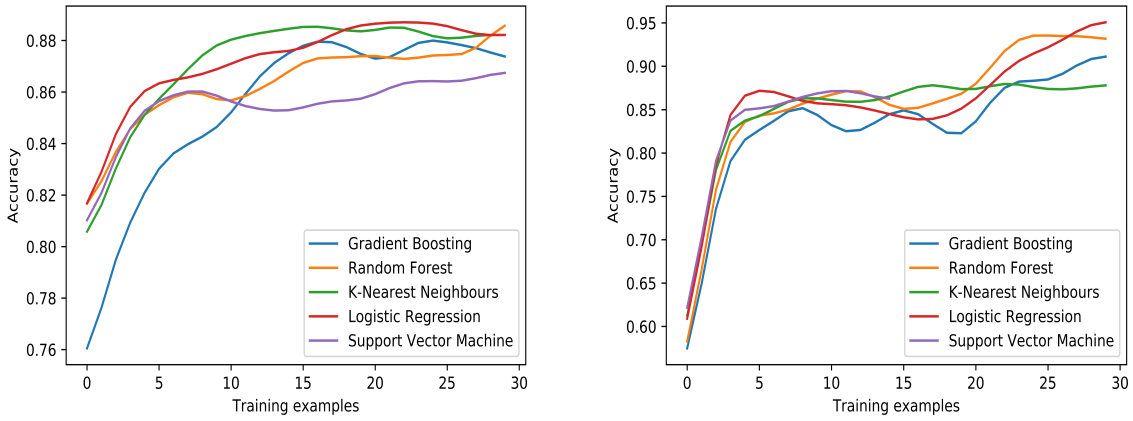
3.6 Results

For both the evaluation strategies mentioned in section 3.5, we take a different number of training example (from 1 to 30) to train the classifier. For a different number of training examples, we observe the accuracy achieved. For both the evaluation strategies, Gradient Boosting, Random Forest, K-Nearest Neighbors, Logistic Regression and Support Vector Machine are trained. K-Nearest Neighbors takes K (nearest neighbors) as a hyper parameter. We also tested K-Nearest



(a) (b)

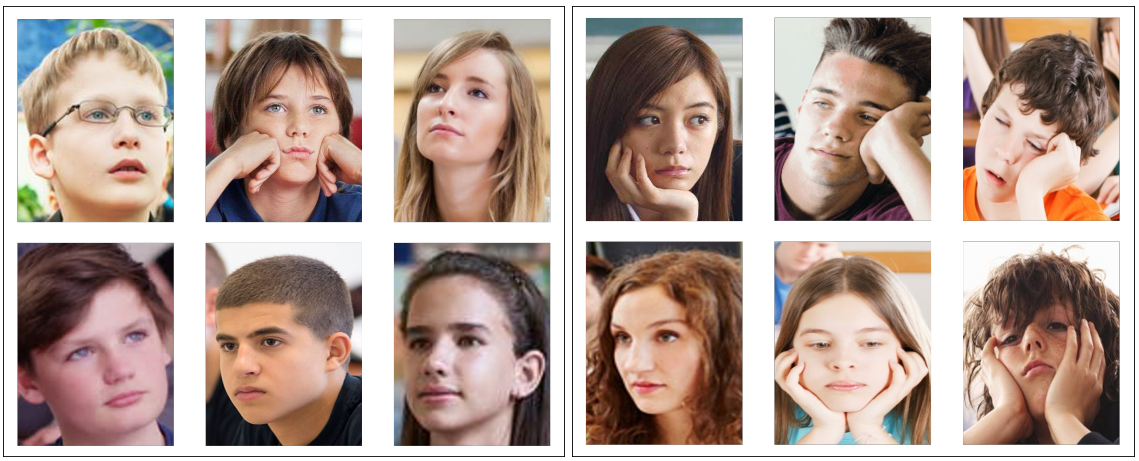
Figure 3.5: K-Nearest Neighbors tested against different number of examples (a) Plot for CK+ dataset, (b) plot for manually labeled new facial expression dataset



(a) (b)

Figure 3.6: Machine learning algorithm tested against different number of examples (a) Plot for CK+ dataset, (b) plot for manually labeled new facial expression dataset. Accuracy on CK+ is achieving accuracy of around 90% which is closer to state-of-the-art accuracy on this dataset (refer table 1.2)

Neighbors with different values of K to get the effect of hyper-parameter on accuracy as shown in figure 3.5. For both the scenario, it is observed that K-Nearest Neighbors works well for K (nearest neighbors) equals 1.



(a) (b)

Figure 3.7: New facial expression recognition output: These examples has two label (new expression), (i) student attentive in class, and (ii) student bored in class. We manually labeled 5 examples of each class and rest automatically labeled examples are shown here. (a) shows attentive student in class labeled by proposed methodology, and (b) shows un-attentive (bored) student in class

Next, the rest of the algorithms are tested. Accuracy plot for both the scenario is shown in figure 3.6. It is observed from both evaluation strategies; Logistic Regression is achieving accuracy best accuracy with as few as 5 examples hence performing better than the rest of algorithms. The results show that as the few examples used for training classifier increases, the accuracy increases up to a certain point and then it saturates. For both the scenario, the number of examples after it saturates is 5.

Figure **??** shows the new expression labeled on face of student in classroom. These examples are collected using user GUI with 2 new facial expression categories, (i) student attentive in class, and (ii) student not attentive in class. Labeled image shows that proposed technique is able to recognizing new expression in un-annotated dataset with as few as 10 examples.

4. CONCLUSION AND FUTURE WORK

4.1 Conclusion

We presented an end solution for recognizing new facial expression using a few labeled examples. Our methodology successfully addresses the issues with limited facial expression databases and absence of context in understanding facial expressions. We also presented a modified object detection model trained specially for detecting faces with greater accuracy and speed. Our face detection model outperforms all other modification (all modification shown in table 2.1) in terms of mean average precision and inference time. We proposed a representation network which is trained on a few basic facial expression databases. This network can recognize subtle expression in the face and generate a distinctive feature vector for that expression. A cascaded classifier able to label them label the new facial expression correctly using these feature vectors from representation network. For both the evaluation criteria, the proposed technique can achieve recognition accuracy of 80% with as few as 5 examples per class. We also presented an analysis for different machine learning algorithms for classifier and observed that logistic regression suits best for classification.

Presented techniques are useful for both developments of FER system for recognizing new facial expression and generation of the context-dependent database. With this technique, any FER system can be trained within milliseconds by training just the end classifier and also using a few examples. We recorded a labeling speed of 38 images per second while generating FER database on Nvidia 1080 Ti GPU. We also developed an easy to use GUI (details and working showed in Appendix A) which can extract context information from the user in the form of support examples and train the classifier for final use.

4.2 Future Work

Since this work has been presented in the form of a methodology for recognizing the new facial expression, there is significant scope for improvement. Specifically, we have identified three areas. Firstly, the whole method is presented in the form of individual modules, and each module

is trained and tested individually. The end-to-end model has the advantage of a single optimization strategy and training, and testing is easy. Secondly, Representation network is chosen as a simple VGG network, which is a functional classification network but not a good discriminator and has a lot of room for improvement in representing face expression as a feature vector. At last, human perceive facial expression in both the spatial and temporal domain. Although presented research uses spatial features to recognize the emotion, accuracy can be much improved by also using temporal features in representing new facial expressions.

REFERENCES

[1] T. Kanade, J. F. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 46–53, 2000.

[2] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[3] P. Ekman, W. V. Friesen, and J. C. Hager, *Facial Action Coding System Investigator's Guide*. Salt Lake City, UT: A Human Face, 2002.

[4] D. Goleman, *Emotional Intelligence*. Bantam Books, 1995.

[5] P. Salovey and J. D. Mayer, "Emotional intelligence," vol. 9, no. 3, pp. 185–211, 1990.

[6] P. Ekman and W. V. Friesen, "Constants across cultures in the face and emotion," *Journal of Personality and Social Psychology*, vol. 17, no. 2, pp. 124–129, 1971.

[7] M. Suwa, N. Sugie, and K. Fujimora, "A preliminary note on pattern recognition of human emotioanl expression," *Proceedings of the 4th International Joint Conference on Pattern Recognition*, pp. 408–410, 1978.

[8] P. Ekman, "Facial expression and emotion," *American Psychologist*, vol. 48, no. 2, pp. 384–392, 1993.

[9] B. M. Lake, R. Salakhutdinov, J. Gross, and J. B. Tenenbaum, "One shot learning of simple visual concepts," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 33, 2011.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *NIPS*, vol. 1, pp. 1097–1105, 2012.

[11] I. S. Pandzica and R. Forchheimer, *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. John Wiley and sons, 2002.

[12] R. Cowie, E. Douglas-Cowie, K. Karpouzis, G. Caridakis, M. Wallace, and S. Kollias, "Recognition of emotional states in natural human-computer interaction," *Multimodal User Interfaces, Springer, Berlin, Heidelberg*, 2008.

[13] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, pp. 94–101, June 2010.

[14] R. Gross, I. Matthews, J. Cohn, T. de, , and S. Baker, "Multipie," *Image and Vision Computing*, vol. 28, no. 5, pp. 807–813, 2010.

[15] M. Pantic, M. Valstar, R. Rademaker, and L. Maat, "Web-based database for facial expression analysis," *IEEE International Conference on Multimedia and Expo, Amsterdam*, p. 5, 2005.

[16] A. Dhall, R. Goecke, J. Joshi, M. Wagner, and T. Gedeon, "Emotion recognition in the wild challenge 2013," in *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*, ICMI '13, (New York, NY, USA), pp. 509–516, ACM, 2013.

[17] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, Y. Zhou, C. Ramaiah, F. Feng, R. Li, X. Wang, D. Athanasakis, J. Shawe-Taylor, M. Milakov, J. Park, R. Ionescu, M. Popescu, C. Grozea, J. Bergstra, J. Xie, L. Romaszko, B. Xu, Z. Chuang, and Y. Bengio, "Challenges in Representation Learning: A report on three machine learning contests," *arXiv e-prints*, p. arXiv:1307.0414, Jul 2013.

[18] C. F. Benitez-Quiroz, R. Srinivasan, and A. M. Martinez, "Emotionet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5562–5570, June 2016.

[19] S. Zafeiriou, A. Papaioannou, I. Kotsia, M. Nicolaou, and G. Zhao, "Facial affect "in-the-wild": A survey and a new database," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1487–1498, June 2016.

[20] A. Mollahosseini, B. Hassani, M. J. Salvador, H. Abdollahi, D. Chan, and M. H. Mahoor, "Facial expression recognition from world wild web," *CoRR*, vol. abs/1605.03639, 2016.

[21] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "Affectnet: A new database for facial expression, valence, and arousal computation in the wild," *IEEE Transactions on Affective Computing*, 2017.

[22] A. Martinez and R. Benavente, "The ar face database," *CVC Technical Report*, vol. 24, 1998.

[23] N. Sebe, T. Gevers, M. S. Lew, Y. Sun, I. Cohen, and T. S. Huang, "Authentic facial expression analysis," *Image and Vision Computing*, vol. 25, no. 12, pp. 1856–1863, 2007.

[24] C. a. Zhang, "A survey of recent advances in face detection," June 2010.

[25] P. Viola and M. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[26] R. B. Girshick, "Fast r-cnn," *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

[27] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[28] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *CoRR*, vol. abs/1506.02640, 2015.

[29] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *CoRR*, vol. abs/1612.08242, 2016.

[30] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018.

[31] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, 2015.

[32] S. Yang, P. Luo, C. C. Loy, and X. Tang, "Faceness-net: Face detection through deep facial part responses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 1845–1859, Aug 2018.

[33] C. Chi, S. Zhang, J. Xing, Z. Lei, S. Z. Li, and X. Zou, "Selective refinement network for high performance face detection," *CoRR*, vol. abs/1809.02693, 2018.

[34] S. Zhang, R. Zhu, X. Wang, H. Shi, T. Fu, S. Wang, T. Mei, and S. Z. Li, "Improved selective refinement network for face detection," *CoRR*, vol. abs/1901.06651, 2019.

[35] V. Jain and E. Learned-Miller, "Fddb: A benchmark for face detection in unconstrained settings," Tech. Rep. UM-CS-2010-009, University of Massachusetts, Amherst, 2010.

[36] S. Yang, P. Luo, C. C. Loy, and X. Tang, "WIDER FACE: A face detection benchmark," *CoRR*, vol. abs/1511.06523, 2015.

[37] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition," vol. 11, no. 4, p. 467, 2002. Exported from https://app.dimensions.ai on 2019/03/28.

[38] C. Shan, S. Gong, and P. W. McOwan, "Facial expression recognition based on local binary patterns: A comprehensive study," *Image Vision Comput.*, vol. 27, pp. 803–816, 2009.

[39] S. M. Mavadati, M. H. Mahoor, K. Bartlett, P. Trinh, and J. F. Cohn, "Disfa: A spontaneous facial action intensity database," *IEEE Transactions on Affective Computing*, vol. 4, pp. 151–160, April 2013.

[40] A. Mollahosseini, D. Chan, and M. H. Mahoor, "Going deeper in facial expression recognition using deep neural networks," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–10, March 2016.

[41] Y. Fan, X. Lu, D. Li, and Y. Liu, "Video-based emotion recognition using cnn-rnn and c3d hybrid networks," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, ICMI '16, (New York, NY, USA), pp. 445–450, ACM, 2016.

[42] Y. Tang, "Deep learning using support vector machines," *CoRR*, vol. abs/1306.0239, 2013.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[44] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *CoRR*, vol. abs/1610.02357, 2016.

[45] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vision*, vol. 115, pp. 211–252, Dec. 2015.

[46] S. Yang, P. Luo, C. C. Loy, and X. Tang, "WIDER FACE: A face detection benchmark," *CoRR*, vol. abs/1511.06523, 2015.

[47] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *CoRR*, vol. abs/1502.01852, 2015.

[48] "Pytorch deep learning library." `https://pytorch.org`. Accessed: 2018-05-01.

[49] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[50] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[51] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

[52] C. Qi and F. Su, "Contrastive-center loss for deep neural networks," *CoRR*, vol. abs/1707.07391, 2017.

APPENDIX A

GUI FOR NEW FACIAL EXPRESSION RECOGNITION

This section presents the details for Graphical User Interface developed for a user for recognizing new facial expression and generating facial expression database with the proposed technique. Figure A.1 shows the GUI interface and explains the basic structure and functions. GUI will load images with a face from a predefined directory. Labels are provided which display the current status of annotation (number of examples labeled by a user for each class (new facial expression) so far). After sufficient labeling, "Train Model" button will train the classifier based on the faces marked by the user.

After starting the labeling, GUI will provide labeling buttons as shown in A.2. Annotation for a particular face, as shown in the right top corner, is done by clicking a corresponding label button and GUI will move to next face in image or next image in the database if the face in the image is over. User can also reject a face if it is not showing correct context with respect to the new expression labels or not properly localized by face detector by clicking "N/A" and the example will not be used in training classifier.

Classifier trained by the GUI based on input provided by the user can be saved and utilized as a FER system for detecting the new facial expression and also for database generation. GUI can further be used to validate the trained classifier by manually observing annotations of new faces in the GUI.

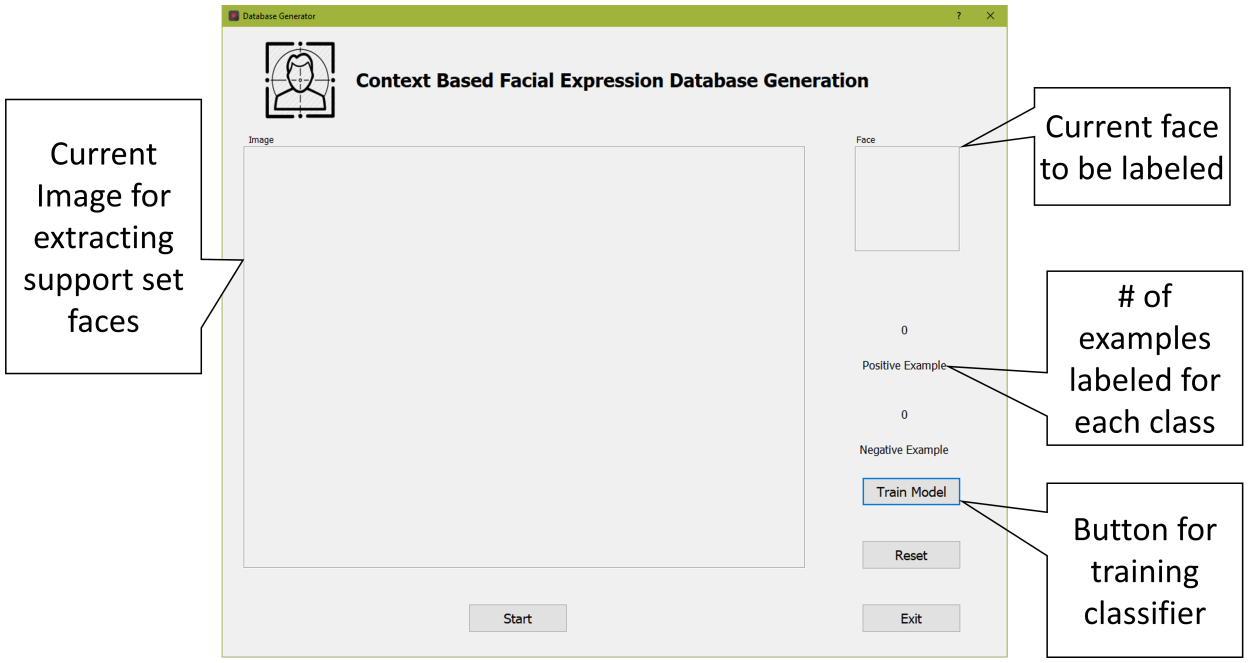


Figure A.1: Interface description

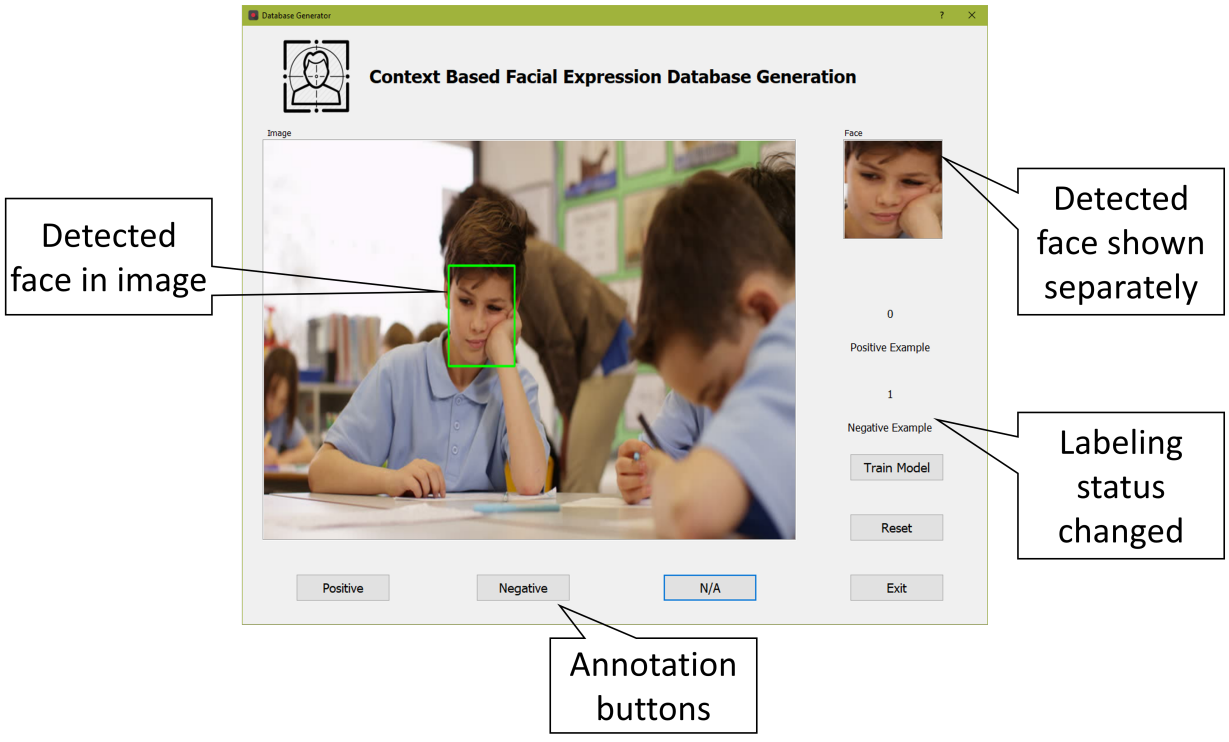


Figure A.2: Annotating new facial expressions